Vectra AI

Version 2.2
REST API Guide

VECTRA

Revision History

DATE	COMMENT
Jan 2021	Release 6.4 • Adds query-parameter caching mechanism to the /api/v2.2/health endpoint
Dec 2020	Release 6.3 • Adds ability to list/add/remove notes for an account via /accounts/<account_id>/notes • Adds ability to list/add/remove notes for a host via /hosts/<host_id>/notes
Nov 2020	Release 6.2 • Adds ability to list linked-account id's with /hosts
Oct 2020	Release 6.2 • Adds tagging support for linked-accounts with /accounts
Sep 2020	Release 6.1 • Add linked-account support to /accounts

API Version 2.2 Changelog

Health

- Adds capability to bypass cached values.

Account

- Updates field ‘account_type’ to be a list of account types.
- Updates field ‘url’ to a v2.2 url for an account.
- Adds capability to list/add/remove notes for an account via /accounts/<account_id>/notes

Host

- Adds capability to list/add/remove notes for a host via /hosts/<host_id>/notes

Overview

A REST API is available for administrators and developers to integrate Vectra’s breach detection data into their applications. Vectra X-series RESTful API provides access to security event data, platform configuration, and health information via URI paths.

Vectra REST API is based on open standards. You can use any web development language to access and retrieve information via the API. A common use-case would be to retrieve security event information generated by the Vectra platform and supply this information to a security operations dashboard or incident response and ticketing systems.

The REST API can be accessed via HTTPS connection to the management interface IP address of the Vectra X-series. The data in the response to the API query is in JSON format.

Examples of security event data that can be integrated into your application:

- Security event type detected
- Host/Account information associated with the security event
- Severity of the Host/Account activities

The Vectra REST API is accessible using token authentication. A token can be generated on the ‘My Profile’ page from the Vectra Dashboard.

Security Detection Data

The "Detections”, “Hosts", and “Accounts” elements retrieve security events that can be inserted into external applications. The REST API provides filtering options to extract data. Advanced parsing of the data can be performed after data has been retrieved and saved into your target application. Order of the response data returned is latest first.

Accessing REST API 2.2

The REST API v 2.2 is accessed via the URL https://<vectra-management-ip>/api/v2.2/<path>.

The <path> options for REST API queries are listed in the table below.

URL	METHOD	DATA TO QUERY
/rules	GET, POST, PUT, DELETE	Triage rule configuration
/detections	GET, PATCH	Security detection events
/accounts	GET	Account
/hosts	GET, PATCH	Host information
/search	GET	Advanced search on hosts and detections
/users	GET, PATCH	User objects
/threatFeeds	GET, POST, DELETE	threatFeed objects
/proxies	GET, POST, PATCH	Proxy objects
/tagging	GET, PATCH	Tags for hosts and detections
/groups	GET, POST, PATCH, DELETE	Groups
/health	GET	System Health information
/lockdown/account	GET	Accounts disabled via Lockdown
/lockdown/host	GET	Hosts disabled via Lockdown
/audits	GET	Audit logs

The REST API is available for all Vectra admin user roles. Version 2.2 uses token authentication. Every user created on the Vectra platform has access to his or her API token under the “My Profile” page. The token provides access to the API based on the user’s role, like UI access. Thus, if a user does not have ability to create triage filters from the UI, that user will also not be able to create triage filters using API.

Example of using token authentication with curl is shown below.

curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94” https://<vectra-management-ip>/api/v2.2/

Triage

Version 2.2 of triage API supports GET, POST, PUT, and DELETE. The API endpoint for accessing version 2.2 is https://<vectra-management-ip>/api/v2.2/rules

The following table lists the top-level fields and descriptions present in the response for a bulk GET request to the v2.2 API for Triage rules.

ELEMENT	DESCRIPTION	TYPE	NOTES
count	The number of object IDs retrieved in the output	integer	
next	URL to the next page of output	string	Useful when using a web-based REST API browser.
previous	URL link to the previous page of output	string	Useful when using a web-based REST API browser.
results	A list of all Triage rules (described in the table below) being returned by the query	list	

The following table lists the fields and descriptions present in a Triage rule. These Triage rules will be contained inside the 'results' field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
id	The ID of the Triage rule	integer	
url	A v2.2 URL to the Triage rule	string	Useful when using a web-based REST API browser.
enabled	Describes if the Triage rule is enabled	boolean	
created_timestamp	Describes the time the Triage rule was created	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
last_timestamp	The timestamp when this Triage filter was triggered	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
is_whitelist	This whitelists all detections for this activity	boolean	True or False
priority	Used in ordering execution of Triage filters	integer	
active_detections	The total number of active detections this Triage rule applies to	integer	
total_detections	The total number of detections (active or inactive) this Triage rule applies to	integer	
template	Specifies if this Triage rule was created based off of a template	Boolean	True of False
detection_category	Original detection category	string	

ELEMENT	DESCRIPTION	TYPE	NOTES
triage_category	Custom Triage label used to categorize specified detections	string	
detection	Original detection type	string	
source_conditions	Specifies the entity this Triage rule applies to	JSON	source_conditions respresents the conditions that can be applied to the source of a detection, including host, account, IP, and sensor. For more information on the format of source_conditions, see below this table
additional_conditions	Specifies additional matching criteria for this Triage rule	JSON	additional_conditions are other conditions that are different on a per-detection-type basis. For more information on the format of additional_conditions, see below this table

Formatting source_conditions and additional_conditions

The format of version 2.2 Triage rules are based on AND/OR logic, making them more flexible and customizable to your specific situation. Both 'source_conditions' and 'additional_conditions' are now saved as JSON blobs which represents a tree-like structure. Each tree node is represented in the following format:

```
{operator: operand}
```

Where "operator" is any of the following:

```
For non-leaf nodes: 'AND' or 'OR'
For leaf nodes: 'ANY_OF' or 'NONE_OF'
```

And "operand" is any of the following:

```
For non-leaf nodes: A list of children tree nodes of the form [{operator: operand}, {operator: operand}, …]
For leaf nodes: A dictionary of the form:
{
    'field': <FIELD NAME>,
    'values': [
        {'value': [<VALUE>]}
    ],
```

```
                'group': [
                        {'value': [<GROUP ID>]}
                ]
}
```

Stipulations on the format of the top-level tree structure as of 2.2 is that the top-level operator must be an 'OR' node, with a single 'AND' node as the only child. The 'AND' node may have an arbitrary number of leaf node children. All valid 2.2 Triage rules will look as follows:

```
{
        'OR': [
                { 'AND': [
                        <LEAF NODE 1>,
                        <LEAF NODE 2>,
                        ...
                        <LEAF NODE N>,
                ]}
        ]
}
```

source_conditions and additional_conditions have different fields that can be used in their leaf nodes. For source_conditions, the following fields are valid: ip, host, account, sensor. The fields 'ip' and 'host' also support triaging on groups, meaning that an ip or a host leaf node can be given an IP Group or Host Group ID, and the Triage rule will apply to every IP/host in the specified group.

For additional_conditions, the following fields are valid: remote1_ip, remote1_proto, remote1_port, remote1_dns, remote2_ip, remote2_proto, remote2_port, remote2_dns, account, named_pipe, uuid, identity, share, extensions, rdp client name, rdp client token, keyboard name. The fields 'remote1_ip', 'remote2_ip', 'remote1_dns', 'remote2_dns' support triaging on groups.

Either source_conditions or additional_conditions may be null. Setting source_conditions to null implies that this Triage rule with apply to All Hosts.

To see examples of complete valid source_conditions and additional_conditions, see Appendix A.

Detections

Detection objects contain all the information related to security events detected on the network. The URL to retrieve all detections is https://<vectra-management-ip>/api/v2.2/detections and uses Token auth.

Version 2.2 introduces listing/adding/updating/removing 'Notes' on a Detection entity via GET, POST, PATCH, DELETE methods. The API endpoint for accessing this in version 2.2 is:
https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes
See: Appendix A: 'Detections (Notes)' section for examples.

Detections are grouped into one of the following categories:

DETECTION CATEGORY	URL
Command & Control	/detections?category=command
Botnet	/detections?category=botnet
Reconnaissance	/detections?category=reconnaissance
Lateral Movement	/detections?category=lateral
Exfiltration	/detections?category=exfiltration
Info	/detections?category=info

The API query performs partial word match. For example, you can use /detections?category=recon to query all Reconnaissance category detections.

An example of using curl to retrieve all detections using token authentication:

curl –H "Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94" https://<vectra-management-ip>/api/v2.2/detections

The following table lists the top-level fields and descriptions present in the response for a bulk GET request to the v2.2 API for Detections

ELEMENT	DESCRIPTION	TYPE	NOTES
count	The number of object IDs retrieved in the output	integer	
next	URL to the next page of output	string	Useful when using a web-based REST API browser.
previous	URL link to the previous page of output	string	Useful when using a web-based REST API browser.
results	The list of Detections returned in the output	List of Objects	

The following table lists the fields and descriptions present in a Detection. These Detections will be contained inside the 'results' field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
id	Object ID	integer	
detection	The name of the threat detected.	string	See Appendix B for the list of Detection names

ELEMENT	DESCRIPTION	TYPE	NOTES
detection_type	The name of the threat detected.	string	See Appendix B for the list of Detection names
category	The category of the vname attack detected.	string	See Appendix B for the list of categories. Will be deprecated in future release. Replaced by detection_category.
detection_category	The category of the vname attack detected.	string	See Appendix B for the list of categories.
src_ip	The source IP address of the host attributed to the security event.	string	
state	The state of the detection.	string	If the detection has aged out the state will transition to “active” from “inactive”. If marked as fixed in the UI, or via the V2.2 API, state will be “fixed”.
t_score	The threat score attributed to the detection.	integer	Will be deprecated in future release. Replaced by threat.
threat	The threat score attributed to the detection.	integer	
c_score	The certainty score attributed to the detection.	integer	Will be deprecated in future release. Replaced by certainty.
certainty	The certainty score attributed to the detection.	Integer	
first_timestamp	The timestamp when the event was first detected	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
last_timestamp	The timestamp when the event was last detected	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
description	System generated description of the event.	string	
proto	Protocol used in the communications.	string	

ELEMENT	DESCRIPTION	TYPE	NOTES
total_bytes_sent	Total bytes sent by the client.	integer	
total_bytes_rcvd	Total bytes received by the client.	integer	
url	The URL that links directly to this record.	string	
sensor_name	The name of sensor where this flow was detected from.	string	
src_host	A dictionary with fields that describe the Host the detection is from	JSON object	
url	The URL that links directly to the detection record	string	
summary	The summary information for the detection	JSON object	See Appendix A for examples
grouped_details	The detection details for the detection	JSON object	See Appendix A for examples
tags	User defined tags added to the detection	List of strings	
targets_key_asset	Indicates whether the detection targets a key asset	boolean	Will be deprecated in future release. Use is_targeting_key_asset.
campaign_summaries	The summaries of campaigns of which this detection is part.	JSON object.	The summaries of campaigns this detection belongs to
is_targeting_key_asset	Indicates whether the detection targets a key asset	boolean	
note	User defined note for this detection.	string	

ELEMENT	DESCRIPTION	TYPE	NOTES
note_modified_by	User name who last modified note	string	
note_modifed_timestamp	The timestamp when note was last modified	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
assigned_to	User named assigned to this detection	string	
assigned_date	The timestamp when user was assigned this detection	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
src_account	A dictionary with fields that describe the Account the detection is from	JSON object	

You can also apply filters to the API response to query for specific elements. The available filter options can be viewed by using OPTIONS instead of GET within the REST client.

The available options and filters for detection set is listed below.

QUERY PARAMETER	DESCRIPTION
fields	Filters objects listed. The available fields are listed in Table 3
page	Page number. Possible values are a positive integer or last
page_size	Page size. Possible values are a positive integer, up to 5000.
ordering	Orders records by last timestamp, threat score and certainty score. The default sorts threat and certainty score in ascending order. Scores can be sorted in descending order by prepending the query with “minus” symbol.
min_id	>= the id provided
max_id	<= the id provided
state	filter by state: active, inactive, ignored, ignored for all
category	filter by the detection category
detection_type	filter by the name of the threat detected.
detection_category	filter by the detection category

QUERY PARAMETER	DESCRIPTION
src_ip	filter by source (ip address)
t_score	filter by threat score
t_score_gte	filter by threat score >= the score provided
threat_score	filter by threat score
threat_gte	filter by threat score >= the score provided
c_score	filter by certainty score
c_score_gte	filter by certainty score >= the score provided
certainty	filter by certainty score
certainty_gte	filter by certainty score >= the score provided
last_timestamp	filter by last timestamp
host_id	filter by id of the host object a detection is attributed to
tags	filter by a tag or a comma-separated list of tags
destination	filter by destination in the detection detail set
proto	filter by the protocol in the detection detail set
is_targeting_key_asset	filter by is_targeting_key_asset: True or False
note_modified_timestamp_gte	filter by note_modified_timestamp >= the timestamp provided: '2019-08-27T20:55:29Z'

Examples of detection queries:

QUERY	COMMENT
https://1.1.1.2/api/v2.2/detections/?ordering=t_score	Retrieves all detections with threat score sorted low to high (ascending).
https://1.1.1.2/api/v2.2/detections/?ordering=-t_score	Retrieves all detections with threat score sorted high to low (descending).
https://1.1.1.2/api/v2.2/detections/?c_score_gte=60	Retrieves all detections with certainty score greater than or equal to 60.
https://1.1.1.2/api/v2.2/detections/?page=2	Retrieves page 2 of detections.
https://1.1.1.2/api/v2.2/detections/?t_score_gte=90&c_score_gte=90	Retrieves all detections with threat &certainty score greater than or equal to 90
https://1.1.1.2/api/v2.2/detections/?tags=RAT	Retrieves all detections with tag value=RAT
https://1.1.1.2/api/v2.2/detections/?tags=RAT,TOR	Retrieves all detections with tag value=RAT, and TOR

Example using a Python script to retrieve "Detections" event data of type "Data Smuggler" and from host 172.16.106.116.

```
import requests
import json
vectra_url = 'https://192.168.51.13/api/v2.2/detections'
headers = {'Content-Type': 'application/json', 'Authorization': 'Token 053662afeb2d02bd3773b93bbc514d4bbb92694d'}
payload = {'detection': 'data smuggler', 'src_ip': '172.16.106.116'}
response = requests.get(url=vectra_url, params=payload, verify=False,
headers=headers)
print(response.json())
```

Detection PCAP API

Version 2.0 of the API supports download of PCAP for a particular detection. Users can perform a GET using the following URL format to download pcap for any detection:

https://<vectra-management-ip>/api/v2.2/detections/<id>/pcap

The download of pcap via API is subject to RBAC rules just like UI access. If a user's role does not permit viewing pcap information, he or she will not be able to access pcap via API as well.

Account

Version 2.2 of accounts API supports GET. The API endpoint for accessing version 2.2 is https://<vectra_management_ip>/api/v2.2/accounts

Version 2.2 introduces listing/adding/updating/removing 'Notes' on an Account entity via GET, POST, PATCH, DELETE methods. The API endpoint for accessing this in version 2.2 is:
https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes
See: Appendix A: 'Account (Notes)' section for examples.

The following table lists the top-level fields and descriptions present in the response for a bulk GET request to the v2.2 API for Accounts

ELEMENT	DESCRIPTION	TYPE	NOTES
count	The number of object IDs retrieved in the output	integer	
next	URL to the next page of output	string	Useful when using a web-based REST API browser.
previous	URL to the previous page of output	string	Useful when using a web-based REST API browser.
results	A list of all Accounts (described in the table below) being returned by the query	list	

The following table lists the fields and descriptions present in an Account. These Accounts will be contained inside the ‘results’ field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
id	The ID of the Account	integer	
url	A v2.2 URL to the Account	string	Useful when using a web-based REST API browser.
name	The name associated with the Account	string	
state	The state of the Account	string	
threat	The threat score attributed to the account	integer	
certainty	The certainty score attributed to the account	Integer	
severity	The severity of this Account	string	Either ‘Low’, ‘Medium’, ‘High’, or ‘Critical’. This is calculated based off of the threat and certainty of the Account
account_type	The method through which this account was discovered	List of strings	Can be one or both of “kerberos” or “o365”
tags	User defined tags added to the account	List of strings	
note	User defined note added to the account	string	
note_modified_by	User name who last modified note	string	
note_modifed_timestamp	The timestamp when note was last modified	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
privilege_level	A number 1-10 to represent how privileged this account is	int	
privilege_category	A string to represent how privileged this account it	string	Either ‘Low’, ‘Medium’, or ‘High’. Privilege levels of 1 and 2 map to ‘Low’. Privilege levels of 3-7 map to ‘Medium’. Privilege levels of 8-10 map to ‘High’
last_detection_timestamp	Last detection activity from this Account	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
detection_set	List of Detections for Account	string	

ELEMENT	DESCRIPTION	TYPE	NOTES
detection_summaries	The summaries of detections attached to this Account.	List of Objects	
ldap	Information about the LDAP server this Account came from (if applicable)	JSON object	

The available options and filters for account set is listed below.

QUERY PARAMETER	DESCRIPTION
fields	Filters objects listed
page	Page number. Possible values are a positive integer or last
page_size	Page size. Possible values are a positive integer up to 5000.
ordering	Orders records by last timestamp, threat score and certainty score. The default out sorts threat and certainty score in ascending order. Scores can be sorted in descending order by prepending the query with “minus” symbol.
name	filter by name
state	filter by state: active or inactive
t_score	filter by threat score
t_score_gte	filter by threat score >= the score provided
c_score	filter by certainty score
c_score_gte	filter by certainty score >= the score provided
tags	filter by a tag or a comma-separated list of tags (returns hosts that contain any of the tags specified), e.g.tags=baz | tags=foo,bar"
all	No filter, return all host objects. Only available in version 2.0 API
min_id	Filter hosts hosts have id greater than or equal to min_id
max_id	Filter hosts hosts have id less than or equal to max_id
note_modified_timestamp_gte	filter by note_modified_timestamp >= the timestamp provided: ‘2019-08-27T20:55:29Z’
privilege_level	filter by exact privilege level of hosts. 1-10
privilege_level_gte	filter hosts that have a privilege level greater than or equal to the supplied number. 1-10
privilege_category	filter hosts by privilege category. Options are ‘low’, ‘medium’ and ‘high’

Hosts

Host information includes data that correlates the Host data to detected security events. This information includes but is not limited to:

- Hostname
- IP address
- Threat score
- Certainty score

URL to retrieve hosts information is https://<vectra_management_ip>/api/v2.2/hosts.

Example using a Python script to retrieve “Hosts” information from hostname “TB5-7” is shown below using token authentication. To retrieve the most current host information, you must issue the request with a “state=active” for the current host information. The host information retrieved will include active detection data attributed to that host. A similar search could have been made against the IP address of the host, using the last_source parameter.

```
import requests
import json
vectra_url = 'https://192.168.22.34/api/v2.2/hosts'
headers = {'Content-Type': 'application/json', 'Authorization': 'Token 053662afeb2d02bd3773b93bbc514d4bbb92694d'}
payload = {'state': 'active', ‘name’: ‘TB5-7’}

response = requests.get(url=vectra_url, params=payload, verify=False, headers=headers)
print(response.json())
```

An example of using curl to retrieve all hosts using token authentication:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
https://<vectra_management_ip>/api/v2.2/hosts
```

The hosts API using token authentication will return hosts that have active detections, active traffic or are marked as key assets. To return all the hosts, use query parameter “all”

Version 2.2 introduces listing/adding/updating/removing ‘Notes’ on a Host entity via GET, POST, PATCH, DELETE methods. The API endpoint for accessing this in version 2.2 is:
https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes
See: Appendix A: ‘Host (Notes)’ section for examples.

The following table lists fields and description of the various elements for the hosts.

ELEMENT	DESCRIPTION	TYPE	NOTES
count	The number of object IDs retrieved in the output	integer	

ELEMENT	DESCRIPTION	TYPE	NOTES
next	URL to the next page of output	string	Useful when using a web-based REST API browser.
previous	URL to the previous page of output	string	Useful when using a web-based REST API browser.
results	List of hosts returned in the output	List of Objects	

The following table lists the fields and descriptions present in a Host. These Hosts will be contained inside the ‘results’ field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
id	Object ID	integer	
new_host_pointer	Indicates whether additional records for this host exists	string	Deprecated in version 2.0 of API
name	The learned hostname	string	
state	The state of this host	string	Possible values are “active” or “inactive”. Active hosts are those with active detections
active_traffic	Flag indicating if the host is exhibiting traffic	boolean	Only used in version 2.0 of host API. Possible values are True or False. Will be deprecated in future release. Replaced by has_active_traffic.
has_active_traffic	Flag indicating if the host is exhibiting traffic	boolean	Only used in version 2.0 of host API. Possible values are True or False.
t_score	The current threat score correlated to this host	integer	Will be deprecated in future release. Replaced by threat.
threat	The current threat score correlated to this host	integer	
c_score	The current certainty score correlated to this host	integer	Will be deprecated in future release. Replaced by certainty.
certainty	The current certainty score correlated to this host	integer	
last_source	Last source IP associated with this host	string	
previous_ips	The previous ips observed for this host	List of strings	Only used in version 2.0 of API
last_detection_timestamp	Last detection activity from this host	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT

ELEMENT	DESCRIPTION	TYPE	NOTES
key_asset	Flag indicating if the host is a key asset	boolean	Only used in version 2.0 of API. Possible values “True” or “False”. Will be deprecated in future release. Replaced by is_key_asset.
is_key_asset	Flag indicating if the host is a key asset	boolean	Only used in version 2.0 of API. Possible values “True” or “False”.
targets_key_asset	Flag indicating if the host has a detection targeting a key asset	Boolean	Only used in version 2.0 of API. Possible values “True” or “False”. Will be deprecated in future release. Replaced by is_targetin_key_asset.
is_targeting_key_asset	Flag indicating if the host has a detection targeting a key asset	boolean	Only used in version 2.0 of API. Possible values “True” or “False”.
probable_owner	Probable owner of the host	string	
tags	User defined tags added to the host	List of strings	
note	User defined note added to the host	string	
note_modified_by	User name who last modified note	string	
note_modifed_timestamp	The timestamp when note was last modified	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
host_artifact_set	List of host artifacts observed for the host	List of Objects	
sensor	Sensor that reported seeing the activity for this host	string	
detection_set	List of Detections for Host	string	
url	The URL to link directly to this host record	string	
detection_summaries	The summaries of detections attached to this host.	List of Objects	
campaign_summaries	The summaries of campaigns of which this host is part.	List of Objects	
assigned_to	User named assigned to this detection	string	
assigned_date	The timestamp when user was assigned this detection	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT

ELEMENT	DESCRIPTION	TYPE	NOTES
privilege_level	A number 1-10 to represent how privileged this account is	int	
privilege_category	A string to represent how privileged this account it	string	Either ‘Low’, ‘Medim’, or ‘High’. Privilege levels of 1 and 2 map to ‘Low’. Privilege levels of 3-7 map to ‘Medium’. Privilege levels of 8-10 map to ‘High’

The available options and filters for hosts set is listed below.

QUERY PARAMETER	DESCRIPTION
fields	Filters objects listed
page	Page number. Possible values are a positive integer or last
page_size	Page size. Possible values are a positive integer up to 5000.
ordering	Orders records by last timestamp, threat score and certainty score. The default out sorts threat and certainty score in ascending order. Scores can be sorted in descending order by prepending the query with “minus” symbol.
name	filter by name
state	filter by state: active or inactive
last_source	filter by last_source (ip address)
t_score	filter by threat score
t_score_gte	filter by threat score >= the score provided
c_score	filter by certainty score
c_score_gte	filter by certainty score >= the score provided
last_detection_timestamp	filter by last_detection_timestamp
tags	filter by a tag or a comma-separated list of tags (returns hosts that contain any of the tags specified), e.g.tags=baz | tags=foo,bar"
key_asset	filter by key asset: True, False
targets_key_asset	filter by hosts targeting key assets: True, False. Only available in version 2.0 API
all	No filter, return all host objects. Only available in version 2.0 API
active_traffic	Filter by hosts that have generated active traffic within the last 2 hours. Only available in version 2.0 API
min_id	Filter hosts hosts have id greater than or equal to min_id
max_id	Filter hosts hosts have id less than or equal to max_id
mac_address	filter by mac address

QUERY PARAMETER	DESCRIPTION
note_modified_timestamp_gte	filter by note_modified_timestamp >= the timestamp provided: '2019-08-27T20:55:29Z'
privilege_level	filter by exact privilege level of hosts. 1-10
privilege_level_gte	filter hosts that have a privilege level greater than or equal to the supplied number. 1-10
privilege_category	filter hosts by privilege category. Options are 'low', 'medium' and 'high'

Search

The search API endpoint allows users to perform advanced search against hosts and detections. All attributes of hosts and detections as described in the above sections are searchable (only the ones exposed in the version 2.0 of the endpoints).

Hosts

The search endpoint for hosts is:

https://<vectra_management_ip>/api/v2.2/search/hosts/?page_size=<number>&&query_string=<query>

where <query> is the query that needs to be performed.

page_size is optional. If specified, the page_size will specify the number of results returned per page. If not specified, the default page_size is 50. The maximum value for page_size is 5000.

Some example queries on hosts:

QUERY	DESCRIPTION
'host.threat:>=50 and host.certainty:>=50'	Find all hosts in the critical quadrant
'host.suspicious_admin_learnings.host_manages. protocols:ssh'	Find hosts that administer other hosts over ssh
'host.suspicious_admin_learnings.managers_of_host. protocols:vnc'	Find hosts that are administered over vnc
'host.tags:"alice"'	Find hosts which match the exact tag "alice"
'host.notes:test'	Find notes that contains the phrase "test"
'host.detection_summaries.summary.dst_ports:389'	Find hosts that have detections that target port 389
'host.last_detection_timestamp:'\[now-1d TO now\]'	Find hosts that have had a detection in the last 24 hours

Accounts

The search endpoint for hosts is:

https://<vectra_management_ip>/api/v2.2/search/accounts/?page_size=<number>&&query_string=<query>

where <query> is the query that needs to be performed.

page_size is optional. If specified, the page_size will specify the number of results returned per page. If not specified, the default page_size is 50. The maximum value for page_size is 5000.

Some example queries on accounts:

QUERY	DESCRIPTION
'account.threat:>=50 and account.certainty:>=50'	Find all Accounts in the critical quadrant
'account.tags:"alice"'	Find Accounts which match the exact tag "alice"
'account.notes:test'	Find notes that contains the phrase "test"
'account.detection_summaries.summary.dst_ports:389'	Find Accounts that have detections that target port 389
'acount.last_detection_timestamp:\[now-1d TO now\]'	Find Accounts that have had a detection in the last 24 hours

Detections

The search endpoint for detections is:

https://<vectra_management_ip>/api/v2.2/search/detections/?page_size=<number>&query_string=<query>

where <query> is the query that needs to be performed.

page_size is optional. If specified, the page_size will specify the number of results returned per page. If not specified, the default page_size is 50. The maximum value for page_size is 5000.

Some example queries on detections:

QUERY	DESCRIPTION
'detection.is_triaged:false and detection.grouped_details.target_domains:snakeoil.biz'	Find all active detections that have a target domain of 'snakeoil.biz'
'detection.grouped_details.dst_ports:445'	Find all detections on dst_port 445
'detection.grouped_details.shares:test'	Find all detections that have targeted share "test"
'detection.grouped_details.user_agent:Safari'	Find all detections that have observed user_agent "Safari"
'detection.custom_detection:Misconfiguration'	Find all triaged detections with name "Misconfiguration"
'detection.last_timestamp:\[now-1d to now\] and (detection.is_triaged:false)'	Find all active detections that had detection activity in the last day

Users

User information includes all data corresponding to user accounts. This information includes but is not limited to:

URL to retrieve users information is https://<vectra_management_ip>/api/v2.2/users/

Example using a Python script to retrieve “Users” information from username “cognito” is shown below using token authentication.

```
import requests
import json
vectra_url = 'https://192.168.51.13/api/v2.2/users'
headers = {'Content-Type': 'application/json', 'Authorization': 'Token 053662afeb2d02bd3773b93bbc514d4bbb92694d'}
payload = {'username': 'vadmin'}
response = requests.get(url=vectra_url, params=payload, verify=False, headers=headers)
print(response.json())
```

An example of using curl to retrieve all hosts using token authentication:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
https://<vectra_management_ip>/api/v2.2/users
```

The following table lists fields and description of the various elements for the Users.

ELEMENT	DESCRIPTION	TYPE	NOTES
count	Number of object IDs retrieved in the output	integer	
next	URL to the next page of output	string	Useful when using a web-based REST API browser.
previous	URL link to the previous page of output	string	Useful when using a web-based REST API browser.
results	List of users returned in the output	List of Objects	

The following table lists the fields and descriptions present in a User. These Users will be contained inside the ‘results’ field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
id	Object ID	integer	
last_login	Last time the user logged in	string	Timestamp format: YYYY-MM-DD HH-MM-SS GMT
username	Account name	string	
email	Email associated with the account	string	
account_type	Account type	string	Local, Special, Limited Time Link, LDAP, TACACS

ELEMENT	DESCRIPTION	TYPE	NOTES
authentication_profile	Name of the User's authentication profile	string	LDAP or TACACS only
role	User's group role	string	

The available options and filters for Users is listed below.

QUERY PARAMETER	DESCRIPTION
username	Filter by username
role	Filter by role
account_type	Filter by account type
authentication_profile	Filter by authentication profile
last_login_gte	Filters for User's that have logged in since the given timestamp

Examples of user queries

QUERY	COMMENT
https://1.1.1.2/api/v2.2/users/?page=2	Retrieves page 2 of users
https://1.1.1.2/api/v2.2/users/?username=cognito	Retrieves all users with the username Cognito
https://1.1.1.2/api/v2.2/users/?role=admin	Retrieves all users with the role of admin
https://1.1.1.2/api/v2.2/users/?account_type=local	Retrieves all users with the account type of local
https://1.1.1.2/api/v2.2/users/?authentication_profile=auth-profile	Retrieves all users with either LDAP or TACACS profiles' names auth-profile
https://1.1.1.2/api/v2.2/users/?last_login_gte='2019-08-27T20:55:29Z'	Retrieves all users that have logged in since 2019-08-27T20:55:29Z

ThreatFeeds

The threatFeeds API can be used to automate the upload of STIX files for threat intelligence matching. The API endpoint can also be used to retrieve the current list of threatFeed objects already configured in the system

To obtain the list of threatFeeds, use the following URL

https://<vectra_management_ip>/api/v2.2/threatFeeds/

To obtain details of any threat feed, use the GET method on following URL
https://<vectra_management_ip>/api/v2.2/threatFeeds/<id>/

To delete the threat feed, use the DELETE method on the following URL

https://<vectra_management_ip>/api/v2.2/threatFeeds/<id>/

where <id> is the id of the threatFeed.

The following table lists the fields and a description of the various elements for the “threatFeeds”. Detailed examples of using GET, POST on threatFeeds is described in Appendix A

ELEMENT	DESCRIPTION	TYPE	NOTES
name	The name of the threat feed	string	
duration	The default duration for which indicators in the threat feed are valid	integer	If the indicator in the STIX file has an expiry date, that will be used. The default duration is used for indicators that don’t have any expiry in the STIX file. Every time the file is updated, the default expiry will be the upload date + duration specified.
category	The category in which the detection will fire if a match is observed with any indicator in the threatFeed	string	Valid values are “cnc”, “lateral” or “exfil”
indicatorType	The default indicatorType to use for the observables in the STIX file	string	Valid values are “Anonymization”, “C2”, “Exfiltration”, “Malware Artifacts”, “Watchlist”. The value specified will be used for observables that don’t have an indicator associated with them in the STIX file
certainty	The default certainty to use for indicators in the STIX file	string	Valid values are “Low”, “Medium” or “High”. The certainty value specified will be used if indicator in the STIX file doesn’t have a certainty associated with it.
data	The data from the STIX file	Json object	If the threatFeed has a STIX file already uploaded, then a GET will return the contents of the STIX file in the “data” object, otherwise the data object will be null

Proxies

The proxies API can be used to manage proxy IP addresses (internal or external) in Cognito. The API can be used to retrieve the current list of proxy IP addresses or to create new proxy objects in Cognito.

To obtain the list of proxies use the following URL

https://<vectra_management_ip>/api/v2.2/proxies/

To obtain details of any proxy, use the following URL
https://<vectra_management_ip>/api/v2.2/proxies/<id>

where <id> is the id of the proxy object.

The following table lists the fields and a description of the various elements for “proxies”. Detailed examples of using GET, POST, PATCH on proxies is described in Appendix A

ELEMENT	DESCRIPTION	TYPE	NOTES
source	Whether the proxy was auto detected by Cognito or was added by user	string	Valid values: “user” or “cognito”
id	The identifier of the proxy object	string	
considersProxy	Whether to consider the object as a proxy or not	string	Proxies auto detected by Cognito which have source value as “Cognito” will always have this field set to “true”. User defined proxies can be “true” or “false”
address	The proxy IP address	string	The IP address for the proxy object

Tagging

The tagging API can be used to manage tags for host and detections in Cognito. The API can be used to retrieve or update the current list of tags for either a host or detection.

To manage tags for a host, use the following URL

https://<vectra_management_ip>/api/v2.2/tagging/host/<id>

where <id> is the id of the host object.

To manage tags for a detection, use the following URL

https://<vectra_management_ip>/api/v2.2/tagging/detection/<id>

where <id> is the id of the detection object.

The following table lists the fields and a description of the various elements for tagging operations. Detailed examples of using GET, PATCH on proxies is described in Appendix A

ELEMENT	DESCRIPTION	TYPE	NOTES
status	Status of Tagging request.	string	Valid values: ‘success’ or ‘failure’

ELEMENT	DESCRIPTION	TYPE	NOTES
tags	List of tags for the host or detection object.	list of strings	When doing a PATCH operation, the object will be updated to match the list of tags provided. To remove tags for a host or detection set tags to an empty list.
message	Error message if operation was unsuccessful.	string	Only present when status is 'failure' of operation is a failure.
Invalid_tags	List of tags which are invalid for the host or detetion object.	list of strings	Only present when status is 'failure'. This will only be returned in the response for a PATCH operation.

Groups

The groups API can retrieve a listing of groups that are defined on the system.

Version 2.2 of groups API supports GET, PATCH, POST, and DELETE to not only query the list of groups, but also create, modify or delete them. The path for accessing groups API in version 2.2 of the API endpoint is https://<vectra_management_ip>/api/v2.2/groups

Examples of creating, modifying or viewing groups using API are also described in Appendix A.

A list of groups can be retrieved using the path:

https://<vectra_management_ip>/api/v2.2/groups

Example curl command for fetching a list of groups:

```
curl –H "Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94"
        https://<vectra_management_ip>/api/v2.2/groups
```

Example curl command for creating a group:

```
curl -X POST –H "Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94"
        https://<vectra_management_ip>/api/v2.2/groups
        -d '{"name": "New Group Name",
           "description": "New Group Description",
           "type": "host",
           "members": [1, 2, https://<vectra_management_ip>/api/v2.2/hosts/3]
        }'
```

The following table lists the fields and a description of the various elements for "groups".

ELEMENT	DESCRIPTION	TYPE	NOTES
id	Object id	integer	
name	The name of the group	string	
type	The type of the group (e.g. host)	string	
description	User-defined description of the group	string	
last_modified_timestamp	The datetime of the last modification to the group	datetime	
last_modified_by	The user that last modified the group	string	
members	List of member objects	array	
rules	List of triage rules that the group is attached to	array	

Health

The health API can retrieve a listing of system health information.

Version 2.2 of health API supports GET. The path for accessing health API in version 2.2 of the API endpoint is https://<vectra_management_ip>/api/v2.2/health

Please note that the health API requires the the view health permission to access the system health data via API.

Examples of viewing system health information using the API are also described in Appendix A.

A list of health information can be retrieved using the path:

https://<vectra_management_ip>/api/v2.2/health

Example curl command for fetching a list of system health information:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
        https://<vectra_management_ip>/api/v2.2/health
```

The following table lists the fields and a description of the various elements for “groups”.

ELEMENT		DESCRIPTION	TYPE	NOTES
network		Network information	object	Updated every 5 minutes
	interfaces	A list of network interfaces on the Brain and Sensors	list of objects	Link speed shown in megabits per second

ELEMENT		DESCRIPTION	TYPE	NOTES
	traffic	Peak traffic values on Brain and Sensors	list of objects	Link speed shown in megabits per second
	vlan_count	Number of vlans observed	integer	
system		System information	object	
	serial_number	Serial number of Brain	string	
	uptime	Time since last power on	string	
	version	System version	object	Includes date of last update, version, mode, and model
memory		System memory	object	
	free_bytes	Memory available for system use	integer	Shown in bytes
	dimm_status	DIMM status and location information	list of objects	
	used_bytes	Memory in use	integer	Shown in bytes
	usage_percentage	Percentage of memory in use	integer	
	total_bytes	Total physical memory available	integer	Shown in bytes
sensors		Sensor information	list of objects	
	status	Pairing status of sensor	string	
	name	Name of the sensor	string	
	serial_number	Serial number of sensor	string	
	package_version	Sensor package version	string	
	last_seen	Timestamp of last seen heartbeat from Sensor	datetime	
	ip_address	IP address of Sensor	string	
	luid	Vectra-unique indentifier for Sensor	string	
	location	Location of sensor	string	
disk		Disk information	object	
	raid_disks_missing	All disks in the RAID are present	object	Optional health status and errors, if any

ELEMENT		DESCRIPTION	TYPE	NOTES
	disk_utliization	Filesystem utilization	object	Includes free, used, total, and usage percentage of filesystem in bytes
	degraded_raid_volume	All healthy disks are being utilized by RAID	object	Optional health status and errors, if any
	disk_raid	Overall RAID health	object	Optional health status and errors, if any
cpu		Processor information	object	
	idle_percent	% of CPU idle	integer	
	user_percent	% of CPU processing tasks	integer	
	nice_percent	% of CPU processing higher prioritized tasks	integer	
	system_percent	% of CPU processing system specific tasks	integer	
hostid		HostID coverage health	object	
	ip_always	Number of IP addresses that are always covered by Host ID when seen on the network	integer	
	ip_sometimes	Number of IP addresses that are sometimes covered by Host ID when seen on the network	integer	
	ip_never	Number of IP addresses that are never covered by Host ID when seen on the network	integer	
	artifact_counts	The number of each type of artifact seen on the network	list of objects	Calculated weekly
power		Power supply information	object	
	status	Status of power supply	string	
	power_supplies	List of power supplies and temperature information	list of objects	Shown in celcius, and optional faults, if any
	error	Power supply health errors		Optional power supply errors, if any

The available options and filters for Health is listed below.

QUERY PARAMETER	DESCRIPTION
cache	True by default. If the response is from cache, it will include an *updated_at* property with the datetime at which the health check was performed. Cached values will be no older than five minutes. Setting cache=false will force a new health check.

Account Lockdown

The lockdown API can retrieve a listing of accounts that have been disabled in Active Directory via Detect's Lockdown function.

Version 2.2 of the lockdown API supports GET to query the list of disabled accounts. The path for accessing lockdown in version 2.2 of the API endpoint is https://<vectra_management_ip>/api/v2.2/lockdown/account.

An example of viewing disabled accounts using the API are also described in Appendix A.

A list of disabled accounts can be retrieved using the path:

https://<vectra_management_ip>/api/v2.2/lockdown/account

Example curl command for fetching a list of disabled accounts:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
        https://<vectra_management_ip>/api/v2.2/lockdown/account
```

The following table lists the fields and a description of the various elements for “lockdown”.

ELEMENT	DESCRIPTION	TYPE	NOTES
lock_date	The datetime the acount was disabled	datetime	
locked_by	The name user that disabled the account	string	
unlock_date	The datetime the acount is to be re-enabled per the configured Lockdown timer	datetime	
account_id	The ID of the Account	integer	
account_name	The name of the Account	string	

Host Lockdown

The lockdown API can retrieve a listing of hosts that have been disabled in Microsoft Defender ATP via Detect's Lockdown function.

Version 2.2 of the lockdown API supports GET to query the list of disabled hosts. The path for accessing lockdown in version 2.2 of the API endpoint is https://<vectra_management_ip>/api/v2.2/lockdown/host.

An example of viewing disabled hosts using the API are also described in Appendix A.

A list of disabled hosts can be retrieved using the path:

https://<vectra_management_ip>/api/v2.2/lockdown/host

Example curl command for fetching a list of disabled hosts:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
        https://<vectra_management_ip>/api/v2.2/lockdown/host
```

The following table lists the fields and a description of the various elements for “lockdown”.

ELEMENT	DESCRIPTION	TYPE	NOTES
lock_date	The datetime the host was disabled	Datetime	
locked_by	The name user that disabled the host	String	
unlock_date	The datetime the host is to be re-enabled per the configured Lockdown timer	Datetime	
host_id	The ID of the Host	Integer	
host_name	The name of the Host	string	

Syslog/Kafka configuration

Remote Syslog and Kafka servers can be configured, not created, through the API. Once a remote server has been configured through the UI, a GET request will return a list of configured servers as a JSON document. This document can be be edited and sent as a POST to change the configuration.

In addition to the server, port, proto, cef, types, and filters keys, the v2 key is a boolean which can be used to configure enhanced details for host and account syslog messages.

Appendix A contains examples of GET and POST requests. Described below, is the configuration of Syslog and Kafka filters that are not available in the UI.

In addition to the three toggle filters that are available in the UI, triaged_detections, info_level_detection, and score_decreases, six additional filters can be configured through only the API. The following table describes each filter.

FILTER	DESCRIPTION	NOTES
Triaged_detections	When false, do not include triaged detection syslog messages	boolean
Info_level_detections	When false, do not include info level detection syslog messages	boolean
Score_decreases	When false, do not include host or account syslog messages when both threat/certainty decrease	boolean
Host_threat_threshold	Set an integer threshold below which host syslog messages will not be sent	integer
Host_certainty_threshold	Set an integer threshold below which host syslog messages will not be sent	Integer
Host_observed_privilege_threshold	Set an integer threshold below which host syslog messages will not be sent	integer
Detection_categories	Detection syslog messages are only sent when included in this list, when the filter is enabled	List of strings
Detection_threat_threshold	Set an integer threshold below which detection syslog messages will not be sent	integer
Detection_certainty_threshold	Set an integer threshold below which detection syslog messages will not be sent	integer

A GET request can be used to obtain a list of currently-configured syslog or kafka servers. The response will show a list of available filters. Filters can be configured by a POST request that includes the entire list of servers, as shown in Appendix A.

Every filter has an `enabled` property that is a boolean indicating whether that filter is active. In addition, some filters require a `value` property that can be set to an integer value for the threshold level of the filter. Finally, detection_categories has a `values` property which is a list of strings to set which categories the filter applies to.

Review the listings in Appendix A for example of using GET and POST to configure Syslog and Kafka messaging.

Audits

Audit information includes data that lists requested accesses to resources. This information includes but is not limited to:

- User
- Message describing action
- Result of action
- Timestamp
- Source IP

URL to retrieve audit information is `https://<vectra_management_ip>/api/v2.2/audits`.

An example of using curl to retrieve all hosts using token authentication:

```
curl –H “Authorization: Token db20f83b33744690e4168e7994c8dd0b53e64f94”
https://<vectra_management_ip>/api/v2.2/audits?start=2020-07-06&end=2020-07-07
```

The audits API using token authentication will return audits in the range {start, end}, inclusive. It expects the dates to be passed in the string form YYYY-MM-DD, interpreted as GMT. When start is not included in the request, it defaults to 0001-01-01. When end is not included, it defaults to 9999-12-31. Leaving out both start and end will return all audits stored on the host, which can be up to 200 MB.

The following table lists fields and description of the returned object.

ELEMENT	DESCRIPTION	TYPE	NOTES
start	The interpreted or assumed start date	string	YYYY-MM-DD, GMT
end	The interpreted or assumed end date	string	YYYY-MM-DD, GMT
audits	List of audits returned in the output	List of Objects	Sorted such that audits[0] will be the oldest audit and audits[-1] will be the latest audit

The following table lists the fields and descriptions present in an audit. These audits will be contained inside the 'audits' field, which is a top level field described in the table above.

ELEMENT	DESCRIPTION	TYPE	NOTES
user	Logged in user	string	
role	Role of user	string	Can be null (e.g. ssh'ing to Vectra Appliance)
version	The version of Vectra running on the Vectra Appliance	string	
dvchost	Hostname of Vectra Appliance accessed	string	If not configured, will be the Vectra Appliance's IP
headend_addr	IP of Vectra Appliance accessed	string	
source_ip	IP of user accessing Vectra Appliance	string	
vectra_timestamp	Timestamp of event	string	The string contains an integer that represents the time in seconds since epoch
message	Describes what happened	string	
result	The status of the action	string	"success", "pending", or "failure"

The available options for audits are listed below.

QUERY PARAMETER	DESCRIPTION
start	(Optional) Start date for audit, inclusive, formatted YYYY-MM-DD, in GMT
end	(Optional) End date for audit, inclusive, formatted YYYY-MM-DD, in GMT

Appendix A

This section will include examples of data retrieved from the REST API 2.2. Due to the amount of data that can be retrieved from a single query, the output examples below show only a snippet of the actual data that can be retrieved.

Note: The information in the following examples was generated in a lab environment. Any reference to IP addresses similar to those used in your environment is purely coincidental.

Triage Rules

GET

URL: https://<vectra_management_ip>/api/v2.2/rules/<id>

Headers:
“Authorization”: “Token <api-key>”

Response:


```
{
   "id": 68,
   "url": "https://1.1.1.1/api/v2.2/rules/68",
   "description": "Expected behavior from these devices",
   "enabled": true,
   "created_timestamp": "2019-08-27T20:55:29Z",
   "last_timestamp": null,
   "is_whitelist": false,
   "priority": null,
   "active_detections": 2,
   "total_detections":3,
   "template": true,
   "additional_conditions": {
      "OR": [
         {
            "AND": [
               {
                  "ANY_OF": {
                     "field": "remote1_port",
                     "values": [
                        {
                           "url": null,
                           "value": "135",
                           "label": "135"
                        }
                     ],
                     "groups": [],
                     "label": "Port"
                  }
               }
            ]
         }
      ]
   },
   "source_conditions": {
      "OR": [
```

```
        {
          "AND": [
            {
              "ANY_OF": {
                "field": "host",
                "values": [],
                "groups": [
                  {
                    "url": "https://192.168.51.36/api/v2.2/groups/8",
                    "value": 8,
                    "label": "Cognito - IPAM"
                  }
                ],
                "label": "Host"
              }
            }
          ]
        }
      ]
    },
    "detection_category": "RECONNAISSANCE",
    "triage_category": "Expected IPAM Behavior",
    "detection": "Internal Darknet Scan"
}
```


POST

This post method is used for creating a Triage Rule. There is also the support for using post to mark a detection(s) as custom.

URL: https://<vectra_management_ip>/api/v2.2/rules

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:


```
{
    "description": "Peer to peer triage rule",
    "detection_category": "COMMAND & CONTROL",
    "triage_category": "Miscategorization",
    "detection": "Peer-to-Peer",
    "is_whitelist": false,
    "additional_conditions": {
      "OR": [
       {
         "AND": [
          {
            "ANY_OF": {
```

```
        "field": "remote1_ip",
        "values": [],
        "groups": [
         {
          "value": 2
         }
        ]
       }
      },
      {
       "ANY_OF": {
        "field": "remote1_dns",
        "values": [
         {
          "value": "test.server.com"
         }
        ],
        "groups": [
         {
          "value": 11
         }
        ]
       }
      }
     ]
    }
   ]
  },
  "source_conditions": {
   "OR": [
    {
     "AND": [
      {
       "ANY_OF": {
        "field": "host",
        "values": [
         {
          "value": 1
         }
        ],
        "groups": [
         {
          "value": 8
         }
        ]
       }
      }
```

```
        ]
       }
      ]
     }
   }
```

The following fields are mandatory:

- “detection_category”: Must be set to the category of the detection – LATERAL MOVEMENT, RECONNAISSANCE, COMMAND & CONTROL, EXFILTRATION, BOTNET, INFO
- “detection”: The detection that must be triaged. Use the detection name as seen in the UI or the “Understanding Vectra Detections” guide
- “triage_category”: The name that will be used for the triaged detection. Only applies if “is_whitelist” is set to 0
- “description”: User defined description for the rule
- “is_whitelist”: A Boolean flag indicating whether to create a “whitelist” or a “track without scores” rule
- source_conditions: conditions that can be applied to the source of a detection. Can be NULL
- additional_conditions: other conditions that are different on a per-detection-type basis. Can be NULL

Response:

Success

The response contains the id of the triage rule if successful:

```
{
  "_meta": {
    "message": "Successfully created triage filter",
    "level": "success"
  },
  "id": 11
}
```

Failure

Failure will result in an error message along with an indication of what was incorrect

```
{
  "_meta": {
    "message": "Invalid field(s) found",
    "level": "error"
  },
  "host": [
    "Invalid host: 5 does not map to a real host"
  ]
}
```

PUT

PUT method can be used to modify existing triage filters. A PUT is a full override, so the format will be the same as POST. In order to use PUT to add additional IPs or Hosts to existing triage filters, first use GET to get the existing triage filter for that id, extend the host or IP list with new data and then do a PUT using the complete list. Just performing a PUT with the new list without including existing IPs or hosts will override the existing configuration.

URL: https://<vectra_management_ip>/api/v2.2/rules/<id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
   "detection_category": "LATERAL MOVEMENT",
   "triage_category": "Susp Rmt Exec - Test",
   "detection": "Suspicious Remote Execution",
   "is_whitelist": 0,
   "description": "put test",
      "additional_conditions": {
       "OR": [
        {
         "AND": [
          {
           "ANY_OF": {
            "field": "remote1_ip",
            "values": [
             {
              "value": "1.1.1.1"
             }
            ],
            "groups": [],
           }
          }
         ]
        }
       ]
      },
      "source_conditions": {
       "OR": [
        {
         "AND": [
          {
           "ANY_OF": {
            "field": "ip",
            "values": [
```

```
                {"value": "1.1.1.1"},
                {"value": "1.2.1.1"},
                {"value": "1.1.3.1"}
              ],
              "groups": []
            }
           }
          ]
         }
        ]
       }

}
```


DELETE

This endpoint is used for deleting existing Triage rules.

URL: https://<vectra_management_ip>/api/v2.2/rules/<id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:


```
{
            "detectionIdLIst": [<detection_id1>, <detection_id2>, …]
}
```


Detections

GET

URL: https://<vectra_management_ip>/api/v2.2/detections

Headers:
“Authorization”: “Token <api-key>”

Response:


```
{
 "last_timestamp": "2019-08-27T20:13:08Z",
 "grouped_details": [
  {
   "num_sessions": 10,
   "protocol": "tcp",
   "last_timestamp": "2019-08-27T20:13:08Z",
   "host_detection": 36,
```

```
    "accounts": [],
    "is_host_detail": true,
    "bytes_received": 3425075,
    "dst_geo": null,
    "src_ip": "1.1.1.1",
    "dst_ips": [
     "35.166.75.118"
    ],
    "grouping_field": "last_timestamp",
    "description": null,
    "is_account_detail": false,
    "dst_ports": [
     8080
    ],
    "account_detection": null,
    "first_timestamp": "2019-08-27T20:13:08Z",
    "dst_geo_lat": null,
    "dst_geo_lon": null,
    "bytes_sent": 141982,
    "target_domains": [
     "www.linkedin.com:443"
    ],
    "account_uid": null
   }
 ],
 "custom_detection": null,
 "is_custom_model": false,
 "detection": "Hidden HTTP Tunnel",
 "detection_type": "Hidden HTTP Tunnel",
 "is_targeting_key_asset": false,
 "note_modified_timestamp": null,
 "c_score": 59,
 "t_score": 10,
 "id": 36,
 "category": "COMMAND & CONTROL",
 "src_ip": "1.1.1.1",
 "detection_category": "COMMAND & CONTROL",
 "note": null,
 "state": "active",
 "sensor": "CfxAb1HK",
 "assigned_date": null,
 "targets_key_asset": false,
 "description": null,
 "tags": [],
 "triage_rule_id": null,
 "first_timestamp": "2019-08-27T20:13:08Z",
 "campaign_summaries": [],
```

```
 "groups": [],
 "detection_url": "https://172.168.51.16/api/v2.2/detections/36",
 "src_account": null,
 "sensor_name": "Vectra X",
 "url": "https:// 172.168.51.16/api/v2.2/detections/36",
 "certainty": 59,
 "is_marked_custom": false,
 "note_modified_by": null,
 "summary": {
  "num_sessions": 10,
  "dst_ips": [
   "35.166.75.118"
  ],
  "bytes_sent": 141982,
  "dst_ports": [
   8080
  ],
  "bytes_received": 3425075
 },
 "threat": 10,
 "assigned_to": null,
 "is_triaged": false,
 "src_host": {
  "is_key_asset": false,
  "threat": 90,
  "name": "IP-1.1.1.1",
  "groups": [],
  "url": "https://192.168.51.36/api/v2.2/hosts/2",
  "ip": "1.1.1.1",
  "certainty": 99,
  "id": 2
 }
}
```


PATCH to add a note to a Detection

URL: https://<vectra_management_ip>/api/v2.2/detections/<id>

Headers:

“Authorization”: “Token <api-key>”

“Content-Type”: “application/json”

Body:


```
{
   "note": “This is a detection”,
}
```

PATCH to mark detections as Fixed/Active in bulk

URL: https://<vectra_management_ip>/api/v2.2/detections/

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:
{
"detectionIdList": [<det_id1>, <det_id2>, …],
“mark_as_fixed”: True/False,
}

Detections (Notes)

GET
URL: https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes

Headers:
“Authorization”: “Token <api-key>”

Response:


```
[
 {
  "id": 1,
  "date_created": "2021-01-11T13:42:43Z",
  "date_modified": null,
  "created_by": "vadmin",
  "modified_by": null,
  "note": "this is a detection note"
 },
 {
  "id": 2,
  … … …
 }
]
```


POST
URL: https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:
{"note":"this is a detection note"}

Response:


```
{
  "id": 2,
  "date_created": "2021-01-11T14:14:10.527603Z",
  "date_modified": null,
  "created_by": "vadmin",
  "modified_by": null,
  "note": "this is a detection note"
}
```


GET
URL: https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes/<detection_id>
Headers:
“Authorization”: “Token <api-key>”
Response:


```
{
  "id": 1,
  "date_created": "2021-01-11T13:54:47.987918Z",
  "date_modified": null,
  "created_by": "vadmin",
  "modified_by": null,
  "note": " this is a detection note "
}
```


PATCH
URL: https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes/<detection_id>
Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:
{"note":"updated note"}

Response:
{

```
 "id": 1,
 "date_created": "2021-01-11T13:47:42Z",
 "date_modified": "2021-01-11T13:57:11Z",
 "created_by": "vadmin",
 "modified_by": "vadmin",
 "note": "updated note"
}
```


DELETE

URL: https://<vectra_management_ip>/api/v2.2/detections/<detection_id>/notes/<detection_id>

Headers:

“Authorization”: “Token <api-key>”

Account

GET

URL: https://<vectra_management_ip>/api/v2.2/accounts

Headers:

“Authorization”: “Token <api-key>”

Response:


```
{
 "id": 18,
 "url": "https://192.168.51.36/api/v2.2/accounts/18",
 "name": "mixitupab.anon.anoncorp.com",
 "state": "inactive",
 "threat": 0,
 "certainty": 0,
 "severity": "Low",
 "account_type": ["kerberos"],
 "tags": [],
 "note": null,
 "note_modified_by": null,
 "note_modified_timestamp": null,
 "privilege_level": null,
 "privilege_category": null,
 "last_detection_timestamp": "2019-08-28T19:05:12Z",
 "detection_set": [
  "https://192.168.51.36/api/v2.2/detections/64"
 ],
 "detection_summaries": [
  {
   "tags": [],
   "detection_type": "Privilege Anomaly: Unusual Host",
   "is_targeting_key_asset": false,
```

```
    "detection_id": 64,
    "detection_url": "https://192.168.51.36/api/v2.2/detections/64",
    "certainty": 0,
    "detection_category": "LATERAL MOVEMENT",
    "summary": {
      "src_accounts": [
        {
          "name": "mixitupab.anon.anoncorp.com",
          "privilege_category": null,
          "privilege_level": null,
          "id": 18
        }
      ],
      "src_hosts": [
        {
          "name": "IP-1.1.1.1",
          "privilege_category": null,
          "privilege_level": null,
          "id": 2
        }
      ],
      "services_accessed": [
        {
          "name": "http/calm.mine.way.grand.com",
          "privilege_category": null,
          "privilege_level": null,
          "id": null
        },
        {
          "name": "http/reply.kiss.art.prompt.com",
          "privilege_category": null,
          "privilege_level": null,
          "id": null
        },
      ]
    },
    "state": "active",
    "threat": 0,
    "assigned_to": null,
    "assigned_date": null,
    "is_triaged": false
   }
  ],
  "ldap": null
}
```

Account (Notes)

GET

URL: https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes

Headers:

“Authorization”: “Token <api-key>”

Response:


```
[
 {
  "id": 1,
  "date_created": "2021-01-11T13:42:43Z",
  "date_modified": null,
  "created_by": "vadmin",
  "modified_by": null,
  "note": "this is an account note"
 },
 {
  "id": 2,
  … … …
 }
]
```


POST

URL: https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes

Headers:

“Authorization”: “Token <api-key>”

“Content-Type”: “application/json”

Body:


```
{"note":"this is a note"}
```


Response:


```
{
 "id": 2,
 "date_created": "2021-01-11T14:14:10.527603Z",
 "date_modified": null,
```

```
 "created_by": "vadmin",
 "modified_by": null,
 "note": "this is a note"
}
```


GET

URL: https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

Response:


```
{
 "id": 1,
 "date_created": "2021-01-11T13:54:47.987918Z",
 "date_modified": null,
 "created_by": "vadmin",
 "modified_by": null,
 "note": " this is an account note "
}
```


PATCH

URL: https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

“Content-Type”: “application/json”

Body:


```
{"note":"updated note"}
```


Response:


```
{
 "id": 1,
 "date_created": "2021-01-11T13:47:42Z",
 "date_modified": "2021-01-11T13:57:11Z",
 "created_by": "vadmin",
 "modified_by": "vadmin",
 "note": "updated note"
}
```


DELETE

URL: https://<vectra_management_ip>/api/v2.2/accounts/<account_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

Host

GET

URL: https://<vectra_management_ip>/api/v2.2/hosts

Headers:
"Authorization": "Token <api-key>"

Response:


```
{
   "id": 1029,
   "name": "insightws07",
   "active_traffic": true,
   "t_score": 90,
   "c_score": 99,
   "last_source": "10.16.6.6",
   "previous_ips": [
      "10.16.12.1",
      "10.16.0.1",
    ],
   "last_detection_timestamp": "2019-08-28T19:05:12Z",
   "key_asset": false,
   "state": "active",
   "targets_key_asset": true,
   "probable_owner": "dkelle",
   "detection_set": [
      "https://10.1.6.10/api/v2.2/detections/1354",
      "https://10.1.6.10/api/v2.2/detections/1365",
      "https://10.1.6.10/api/v2.2/detections/1380",
      "https://10.1.6.10/api/v2.2/detections/1388",
      "https://10.1.6.10/api/v2.2/detections/1410",
      "https://10.1.6.10/api/v2.2/detections/1435",
      "https://10.1.6.10/api/v2.2/detections/1436",
      "https://10.1.6.10/api/v2.2/detections/1476"
   ],
   "host_artifact_set": [
      {
         "type": "kerberos",
         "value": "insightws07"
      }
   ],
   "sensor": null,
   "tags": [],
   "note": null,
   "url": "https://10.1.6.10/api/v2.2/hosts/1029"
}
```

PATCH to set or remove the key asset flag on a host

URL: https://<vectra_management_ip>/api/v2.2/hosts/<id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:
```
{
   "key_asset": "True",
}
```

Host (Notes)

GET
URL: https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes

Headers:
“Authorization”: “Token <api-key>”

Response:
```
[
 {
   "id": 1,
   "date_created": "2021-01-11T13:48:31Z",
   "date_modified": null,
   "created_by": "vadmin",
   "modified_by": null,
   "note": "create this note"
 },
 {
   "id": 2,
   … … …
 },
]
```

POST
URL: https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:
```
{"note":"this is a note"}
```

Response:


```
{
 "id": 2,
 "date_created": "2021-01-11T13:54:47.987918Z",
 "date_modified": null,
 "created_by": "vadmin",
 "modified_by": null,
 "note": "this is a note"
}
```


GET

URL: https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

Response:


```
{
 "id": 1,
 "date_created": "2021-01-11T13:54:47.987918Z",
 "date_modified": null,
 "created_by": "vadmin",
 "modified_by": null,
 "note": "another note"
}
```


PATCH

URL: https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

“Content-Type”: “application/json”

Body:


```
{"note":"updated note"}
```


Response:


```
{
 "id": 1,
 "date_created": "2021-01-11T13:47:42Z",
 "date_modified": "2021-01-11T13:57:11Z",
 "created_by": "vadmin",
 "modified_by": "vadmin",
 "note": "updated note"
}
```


DELETE

URL: https://<vectra_management_ip>/api/v2.2/hosts/<host_id>/notes/<note_id>

Headers:

“Authorization”: “Token <api-key>”

Users

GET

URL: https://<vectra_management_ip>/api/v2.2/users/

Headers:

“Authorization”: “Token <api-key>”

Response:


```
{
   "count": 3,
   "next": null,
   "previous": null,
   "results": [
      {
         "id": 1,
         "last_login": "2019-08-27T20:55:29Z ",
         "username": "admin",
         "email": "ad@vectra.ai",
         "account_type": "local",
         "authentication_profile": null,
         "role": "Super Admin"
      },
      {
         "id": 2,
         "last_login": "2019-08-27T20:55:29Z ",
         "username": "vadmin",
         "email": "vadmin@vectra.ai",
         "account_type": "TACACS",
         "authentication_profile": tacacs-profile,
         "role": "Admin"
      },
      {
         "id": 3,
         "last_login": "2019-08-27T20:55:29Z ",
         "username": "cognito",
         "email": "cognito@vectra.ai",
         "account_type": "LDAP",
         "authentication_profile": ldap-profile,
```

```
        "role": "Security Analyst"
      }
    ]
}
```


PATCH to change a user to TACACS

URL: https://<vectra_management_ip>/api/v2.2/users/<id>

"Authorization": "Token <api-key>"
"Content-Type": "application/json"

Body:


```
{
   "account_type": "TACACS",
   "authentication_profile": "tacacs-profile",
}
```


threatFeeds

GET

URL: https://<vectra_management_ip>/api/v2.2/threatFeeds/<id>/

Headers:
"Authorization": "Token <api-key>"

Response:


```
{
   "name": "test",
   "data": null,
   "_rev": "1-69a246ab1d2e970732ea34e195a47486",
   "uploadResults": null,
   "lastUpdatedBy": "admin",
   "lastUpdated": "2017-11-07T18:37:02.517125+00:00",
   "version": "3.11-240-g3ccd27c",
   "_id": "4de02b1bad3f4ff028847c30ef74d5f7",
   "type": "STIX",
   "defaults": {
      "duration": 7,
      "category": "cnc",
      "certainty": "High",
      "indicatorType": "Anonymization"
   },
   "uploadDate": "2017-11-07T18:37:02.517125+00:00"
}
```

POST to create a threatFeed

URL: https://<vectra_management_ip>/api/v2.2/threatFeeds/

Headers:
“Authorization”: “Token <api-key>”

Body:

```
{
   "threatFeed": {
        "name":"test-feed",
         "defaults": {
            "certainty":"High",
            "duration":7,
            "indicatorType":"Anonymization",
            "category":"cnc"
         }
    }
}
```

All the fields shown above are mandatory. See the section on threatFeeds for possible values each label can take.

POST to add or replace STIX file to an existing threatFeed

URL: https://<vectra_management_ip>/api/v2.2/threatFeeds/<id>/

Headers:
“Authorization”: “Token <api-key>”

Body: Send the STIX file as a multi part form data

```
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="file"; filename="stix_test_url_anon.xml"
Content-Type: text/xml


------WebKitFormBoundary7MA4YWxkTrZu0gW—
```

Response Success:

```
{
   "threatFeed": {
      "uploadResults": {
         "category": "cnc",
         "certainty": "High",
```

```
        "expiration": "2019-08-27T20:55:29.27Z ",
        "indicatorType": "Anonymization",
        "observableCount": 4
      }
   }
}
```


If you are using a tool like postman, perform the POST with the body of the POST as “form-data” with Key as “file” and value as the selected file from the file system. Do not add a content type header explicitly for postman, since the tool does it automatically.

proxy

GET

URL: https://<vectra_management_ip>/api/v2.2/settings/proxy

Headers:
“Authorization”: “Token <api-key>”

Response if a proxy is configured:


```
{
   "proxy": {
        "host": "proxy.com",
        "enable": true,
        "port": "8888",
        "authentication": {
              "enable": false,
              "user": ""
        }
   }
}
```


Response if a proxy is not configured:


```
{
   "proxy": {
        "host": "",
        "enable": false,
        "port": "",
        "authentication": {
              "enable": false,
              "user": ""
        }
   }
}
```

proxies

GET

URL: https://<vectra_management_ip>/api/v2.2/proxies/

Headers:
“Authorization”: “Token <api-key>”

Response:

```
{
  "meta": {
    "count": 3
  },
  "proxies": [
    {
      "source": "user",
      "id": "f62afac59c994b9c9c18d04b1e1095c4",
      "considerProxy": true,
      "address": "1.2.3.5"
    },
    {
      "source": "user",
      "id": "57c9042358a64cbaaf13dc1a2e0b43b9",
      "considerProxy": true,
      "address": "10.1.2.5"
    },
    {
      "source": "user",
      "id": "1955087a53cb4f3aa825a405a7024b34",
      "considerProxy": true,
      "address": "1.2.3.7"
    }
  ]
}
```

POST to create a proxy

URL: https://<vectra_management_ip>/api/v2.2/proxies/

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
 "proxy": {
  "address": "1.2.3.8",
```

```
   "considerProxy": true
  }
}
```

All the fields shown above are mandatory.

PATCH to modify a proxy object

URL: https://<vectra_management_ip>/api/v2.2/proxies/<id>/

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
  "proxy": {
    "address": "1.2.3.9",
    "considerProxy": true
  }
}
```

Patch operation can only be used for user defined proxies, not for proxy objects that have the source as “Cognito”.

tagging

GET to retrieve tags for a host

URL: https://<vectra_management_ip>/api/v2.2/tagging/host/<host_id>

Headers:
“Authorization”: “Token <api-key>”

Response:

```
{
  "status": "success",
  "tag_id": "1000",
  "tags": [
    "test",
    "this is a tag",
    "We need to follow up on this host."
  ]
}
```

PATCH to add “new_tag” for a host

URL: https://<vectra_management_ip>/api/v2.2/tagging/host/<host_id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
  "tags": [“test”, “new_tag”, “this is a tag”, “We need to follow up on this host.”]
}
```

PATCH to clear tags for a detection

URL: https://<vectra_management_ip>/api/v2.2/tagging/detection/<detection_id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
  "tags": []
}
```

Patch operation will alter the tags for the host or detection to match the “tags” list provided in the PATCH.

Groups

GET

URL: https://<vectra_management_ip>/api/v2.2/groups/6

Headers:
“Authorization”: “Token <api-key>”

Response:

```
{
   "id": 6,
   “name”: “Scanner Group”,
   “description”: “Custom Scanner Group”,
   “last_modified”: “2019-01-25T21:57:14.789794Z”,
   “last_modified_by”: “admin”,
   “type”: “host”,
```

```
  “members”: [
    {
      “url”: https://<vectra_management_ip>/api/v2.2/hosts/161,
      “is_key_asset”: false,
      “id”: 161,
      “name”: “IP-10.16.12.1”
    }
  ],
  “rules”: []
}
```


POST

URL: https://<vectra_management_ip>/api/v2.2/groups/<id>

Headers:

“Authorization”: “Token <api-key>”

“Content-Type”: “application/json”

Body:


```
{
        “name”: “New Group Name”,
        “description”: “New description text”,
        “type”: “host”,
        “members”: [1, 2, “https://<vectra_management_ip>/api/v2.2/hosts/161”]
}
```


The following fields are mandatory:

- “name”: group name is required and must be unique
- “type”: group type is required, “host” is one possible type of group
- “description”: group description is required
- “members”: zero or more members are allowed

Response:

Success

The response contains the id of the group if successful:


```
{
        “group”: {
                “id”: 52
        }
}
```


Failure

Failure will result in an error message along with an indication of what was incorrect. Shown below is the error when trying to create a group without specifying a name.

```
{
  "_meta": {
    "message": "Invalid field(s) found",
    "level": "error"
  },
  "name": [
    "This field may not be null."
  ]
}
```


PATCH

PATCH method can be used to modify existing groups. A PATCH is a partial override, so the format can include one or more fields from the POST. In order to use PATCH to modify the members, a complete new list must be included. For example, if a group already contains hosts 1, 2, 3, 4, and the group should be modified to include host 5, then 5 should be added to the list, the members field of the PATCH request should be 1, 2, 3, 4, 5.

URL: https://<vectra_management_ip>/api/v2.2/groups/<id>

Headers:
“Authorization”: “Token <api-key>”
“Content-Type”: “application/json”

Body:

```
{
        “name”: “New Group Name”,
        “description”: “New description text”,
        “type”: “host”,
        “members”: [1, 2, “https://<vectra_management_ip>/api/v2.2/hosts/161”]
}
```

DELETE

URL: https://<vectra_management_ip>/api/v2.2/groups/<id>

Headers:
“Authorization”: “Token <api-key>”

Health

GET

URL: https://<vectra_management_ip>/api/v2.2/health

Headers:
“Authorization”: “Token <api-key>”

Response:

```
{
  "network": {
    "interfaces": {
      "brain": {
        "eth0": {
          "speed_mbps": 1000,
          "duplex": "FULL",
          "link": "UP"
        },
        "eth1": {
          "link": "DOWN"
        }
      },
      "sensors": {}
    },
    "traffic": {
      "brain": {
        "interface_peak_traffic": {
          "eth3": {
            "peak_traffic_mbps": 0
          },
          "eth2": {
            "peak_traffic_mbps": 0
          },
          "eth1": {
            "peak_traffic_mbps": 0
          },
          "eth0": {
            "peak_traffic_mbps": 0
          }
        },
        "aggregated_peak_traffic_mbps": 0
      },
      "sensors": {}
    },
    "vlan_count": 0
  },
  "system": {
    "serial_number": "S11181714785673",
    "uptime": "35 days, 1 hours, 5 minutes",
    "version": {
      "last_update": "Fri Nov 22 20:07:40 2019",
      "mode": "mixed",
      "vectra_version": "5.3.0-0-4742",
      "model": "X4"
    }
  },
```

```
"memory": {
   "free_bytes": 47405064192,
   "dimm_status": [
      {
         "status": "OK",
         "dimm": "mc1"
      },
      {
         "status": "OK",
         "dimm": "mc0"
      }
   ],
   "used_bytes": 87754235904,
   "usage_percent": 39.5,
   "total_bytes": 135159300096
},
"sensors": [
   {
      "status": "paired",
      "name": "Test Sensor APP Team",
      "serial_number": "V422ry78u5q5673s8koq4orxc8fb96ac6",
      "package_version": "4.9.0-13-31",
      "last_seen": "2019-11-26T16:16:39Z",
      "ip_address": "192.168.5.238",
      "luid": "mnbtl29o",
      "location": "None"
   }
],
"disk": {
   "raid_disks_missing": {
      "status": "OK",
      "output": {},
      "error": ""
   },
   "disk_utilization": {
      "free_bytes": 34955042816,
      "total_bytes": 62861103104,
      "usage_percent": 44.39,
      "used_bytes": 27906060288
   },
   "degraded_raid_volume": {
      "status": "OK",
      "output": [],
      "error": ""
   },
   "disk_raid": {
      "status": "OK",
```

```
      "output": "",
      "error": ""
    }
  },
  "cpu": {
    "idle_percent": 66.8,
    "user_percent": 32.5,
    "nice_percent": 0,
    "system_percent": 0.5
  },
  "hostid": {
    {
      "ip_always": 0,
      "ip_sometimes": 0,
      "ip_never": 0,
      "artifact_counts": {
        "proxy_ip": 0,
        "cookie": 0,
        "kerberos": 0,
        "clear_state": 0,
        "dns": 0,
        "crowdstrike": 0,
        "idle_end": 0,
        "src_port": 0,
        "split": 0,
        "vmachine_info": 0,
        "kerberos_user": 0,
        "arsenic": 0,
        "end_time": 0,
        "mdns": 0,
        "idle_start": 0,
        "uagent": 0,
        "static_ip": 0,
        "carbon_black": 0,
        "dhcp": 0,
        "netbios": 0,
        "total": 0,
        "rdns": 0
      }
    }
  },
  "power": {
    "status": "OK",
    "power_supplies": {
      "1": {
        "faults": [],
        "temperature_celcius": 31.0
```

```
        }
      },
      "error": ""
   }
}
```


Account Lockdown

GET

URL: https://<vectra_management_ip>/api/v2.2/lockdown/account

Headers:
“Authorization”: “Token <api-key>”

Response:


```
{
   "lock_date": "2020-02-21T15:59:24Z",
   "locked_by": "admin",
   "unlock_date": "2020-02-21T16:59:24Z",
   "account_id": 79,
   "account_name": "Lockdown_User_1@redwoods.test"
   “type”: “host”,
}
```


Host Lockdown

GET

URL: https://<vectra_management_ip>/api/v2.2/lockdown/host

Headers:
“Authorization”: “Token <api-key>”

Response:


```
[
 {
   "host_id": 10,
   "locked_by": "vadmin",
   "unlock_date": "2020-06-09T15:21:06Z",
   "host_name": "def-1",
   "lock_date": "2020-06-09T14:21:06Z"
 }
]
```

Syslog/Kafka

GET

URL: https://<vectra_management_ip>/api/v2.2/settings/syslog_config

Headers:
“Authorization”: “Token <api-key>”

Response:


```
{
  "syslog_servers": [
    {
      "cef": "cef",
      "proto": "udp",
      "server": "<server_ip>",
      "filters": {
        "info_level_detections": {
          "enabled": false
        },
        "detection_categories": {
          "values": [],
          "enabled": false
        },
        "detection_certainty_threshold": {
          "enabled": false,
          "value": 0
        },
        "detection_threat_threshold": {
          "enabled": false,
          "value": 0
        },
        "triaged_detections": {
          "enabled": false
        },
        "host_certainty_threshold": {
          "enabled": false,
          "value": 0
        },
        "score_decreases": {
          "enabled": false
```

```
            },
            "host_threat_threshold": {
               "enabled": false,
               "value": 0
            },
            "host_observed_privilege_threshold": {
               "enabled": false,
               "value": 0
            }
         },
         "port": 514,
         "types": [
            "host",
            "detection"
         ],
         "v2": false,
      },
      {},
      {}
   ]
}
```


GET

URL: https://<vectra_management_ip>/api/v2.2/settings/kafka_config

Headers:
“Authorization”: “Token <api-key>”

Response:


```
{
   "kafka_servers": [
      {
         "cef": "cef",
         "proto": "udp",
         "bootstrap_servers": "<server_ip>:<port>",
         "filters": {
            "info_level_detections": {
               "enabled": false
            },
```

```
      "detection_categories": {
        "values": [],
        "enabled": false
      },
      "detection_certainty_threshold": {
        "enabled": false,
        "value": 0
      },
      "detection_threat_threshold": {
        "enabled": false,
        "value": 0
      },
      "triaged_detections": {
        "enabled": false
      },
      "host_certainty_threshold": {
        "enabled": false,
        "value": 0
      },
      "score_decreases": {
        "enabled": false
      },
      "host_threat_threshold": {
        "enabled": false,
        "value": 0
      },
      "host_observed_privilege_threshold": {
        "enabled": false,
        "value": 0
      }
    },
    "port": 514,
    "types": [
      "host",
      "detection"
    ],
    "v2": false,
  },
```

```
    {},
    {}
  ]
}
```

POST

URL: https://<vectra_management_ip>/api/v2.2/settings/syslog_config

Headers:
“Authorization”: “Token <api-key>”

BODY/json:

```
{
  "syslog_servers": [
    {
      "cef": "cef",
      "proto": "udp",
      "server": "<server_ip>",
      "filters": {
        "info_level_detections": {
          "enabled": false
        },
        "detection_categories": {
          "values": [],
          "enabled": false
        },
        "detection_certainty_threshold": {
          "enabled": false,
          "value": 0
        },
        "detection_threat_threshold": {
          "enabled": false,
          "value": 0
        },
        "triaged_detections": {
          "enabled": false
        },
        "host_certainty_threshold": {
          "enabled": false,
          "value": 0
        },
        "score_decreases": {
          "enabled": false
```

```
            },
            "host_threat_threshold": {
               "enabled": false,
               "value": 0
            },
            "host_observed_privilege_threshold": {
               "enabled": false,
               "value": 0
            }
         },
         "port": 514,
         "types": [
            "host",
            "detection"
         ]
         "v2": true,
      },
      {},
      {}
   ]
}
```


POST

URL: https://<vectra_management_ip>/api/v2.2/settings/kafka_config

Headers:
“Authorization”: “Token <api-key>”

BODY/json:


```
{
   "syslog_servers": [
      {
         "cef": "cef",
         "proto": "udp",
         "bootstrap_servers": "<server_ip>:<port>",
         "filters": {
            "info_level_detections": {
               "enabled": false
            },
            "detection_categories": {
               "values": [],
               "enabled": false
            },
            "detection_certainty_threshold": {
               "enabled": false,
               "value": 0
```

```
            },
            "detection_threat_threshold": {
               "enabled": false,
               "value": 0
            },
            "triaged_detections": {
               "enabled": false
            },
            "host_certainty_threshold": {
               "enabled": false,
               "value": 0
            },
            "score_decreases": {
               "enabled": false
            },
            "host_threat_threshold": {
               "enabled": false,
               "value": 0
            },
            "host_observed_privilege_threshold": {
               "enabled": false,
               "value": 0
            }
         },
         "types": [
            "host",
            "detection"
         ],
         "v2": true,
      },
      {},
      {}
   ]
}
```


Appendix B

Detections

Predefined categories and vnames

CATEGORY	VNAME
Command and Control	
	External Remote Access
	Hidden DNS Tunnel

CATEGORY	VNAME
	Hidden HTTP Tunnel
	Hidden HTTPS Tunnel
	Malware Update
	Peer to Peer
	Pulling Instructions
	Stealth HTTP Post
	Suspect Domain Activity
	Suspicious HTTP
	TOR Activity
	Threat Intelligence Match
	Suspicious Relay
	Multi-home Fronted Tunnel
Botnet	
	Abnormal Ad Activity
	Abnormal Web Activity
	Cryptocurrency Mining
	Brute-Force
	Outbound DoS
	Outbound Port Sweep
	Outbound Spam
Reconnaissance	
	File Share Enumeration
	Internal Darknet Scan
	Kerberos Account Scan
	Port Scan
	Port Sweep
	SMB Account Scan
	RDP Recon
	RPC Recon
	Suspicious LDAP Query
Lateral Movement	
	Automated Replication
	Brute-Force
	Kerberos Brute-Force

CATEGORY	VNAME
	Kerberos Server Access
	Privilege Anomaly: Unusual Account on Host
	Privilege Anomaly: Unusual Host
	Privilege Anomaly: Unusual Service
	Privilege Anomaly: Unusual Service from Host
	Privilege Anomaly: Unusual Trio
	Ransomware File Activity
	Shell Knocker Client
	Shell Knocker Server
	SMB Brute-Force
	SQL Injection Activity
	Suspicious Admin
	Suspicious Kerberos Account
	Suspicious Kerberos Client
	Suspicious Remote Desktop
	Suspicious Remote Execution
	Threat Intelligence Match
Exfiltration	
	Data Smuggler
	Hidden DNS Tunnel
	Hidden HTTP tunnel
	Hidden HTTPS tunnel
	Smash and Grab
	Staged Transfer - Hop 1
	Threat Intelligence Match
Info	
	Custom

Settings

Predefined group values.

GROUP POSSIBLE KEY VALUES	DESCRIPTION OF THE VALUES
vectra	Lists the hostname assigned to the Vectra brain

GROUP POSSIBLE KEY VALUES	DESCRIPTION OF THE VALUES
dns	Lists the DNS servers configured for host resolution. Up to 3 servers can be configured.
ntp	Lists the NTP server used for time synchronization. Up to 5 servers can be configured.
syslog	Lists the syslog servers and their configuration used for log forwarding. Up to 6 servers can be configured.
platform	Lists the OS version installed.
digest	Lists the email addresses to forward reports to. Up to 3 email addresses can be added to receive the digest emails.
alert	Lists the email addresses to forward detection alerts to. Up to 3 email addresses can be added to receive the alert emails.
smtp	Lists the SMTP server information used to relay alert and digest emails from.
datasharing	Lists the current setting to share data to the Vectra cloud. The key value is boolean, either True or False.
update	Lists the timestamp when the last OS update was applied to the system.
feature	Lists the setting for the community feature to be enabled or disabled. The key value is boolean, either On or Off.